We greatly appreciate the reviewers for taking a close look at the paper and the proofs, and giving a detailed feedback. We first address the common concerns raised.

**Dependence on \(d\) in Lemma 1:** While our focus has been on the dependence of our algorithm on the number of plug-in calls, we understand why the reviewers would like the dependence on \(d\) to be made explicit. Below, we expand the statistical error term in Lemma 1 to show the dependence on \(d\), and will include this in the paper along with the complete proof. This is the same dependence that the previous method of Narasimhan (2018) incurs [26].

For a \(n\)-class problem, let \(\tilde{g}^a, \tilde{u}^a = \text{plug-in}(a)\) as in Algorithm 1. Then with probability \(\geq 1 - \delta\) over draw of \(N\) examples from the data distribution, we have for all \(a \in \mathbb{R}^d\):

\[
\|C[\tilde{g}^a] - \tilde{u}^a\|_2 = O \left( d \sqrt{\frac{\log(d) + \log(Nn^2) + \log(1/\delta)}{N}} \right)
\]

where the notation \(O\) only hides absolute constants. The proof follows from a straightforward application of a result from Cesa-Bianchi & Haussler (1998) to bound the growth function.

**Proof of Proposition 2:** Proposition 2 is straightforward and simply follows from expanding \(\langle a, C[h]\rangle\) as \(\mathbb{E}_X \left[ \sum_{y=1}^n \eta_y(X) \sum_{i=1}^d a_i \sigma_i(X, y, h(X)) \right]\). Hence the Bayes-optimal classifier \(h\) predicts for any given \(x\), a label \(\hat{y}\) that minimizes the inner term \(\sum_{y=1}^n \eta_y(x) \sum_{i=1}^d a_i \sigma_i(x, y, \hat{y})\), i.e. \(h(x) = \arg\min_{\hat{y} \in [n]} \sum_{y=1}^n \eta_y(x)L_{y, \hat{y}}(x)\).

We’ll definitely include this in the appendix.

**Reviewer 2: Lipschitzness.** The fairness and coverage constraints in Section 2 are Lipschitz in the confusion matrix, and so are the H-mean, Q-mean and Min-max metrics. For the G-mean and KLD metrics, we can easily construct close-approximations that are Lipschitz. We’ll include these details in the paper, along with an example. As for the parameter \(\lambda\), in theory it is sufficient to set it to a large-enough value as specified in Lemma 7. In practice, we set \(\lambda = 10\), but the results were robust to changes in \(\lambda\). Thanks for the suggestions to improve the writing and pointing out the typos!

**Reviewer 3: Limitations of a pre-fixed classifier.** We agree that the performance of a plug-in classifier depends on the quality of the base class probability model. As an alternative, one can always train a new classifier from scratch in each step of Algorithm 1 to solve the linear minimization (LMO) over \(C\) (line 5). This amounts to solving a cost-sensitive learning problem at each step. While the modified algorithm will be computationally more expensive, it no longer depends on a pre-trained model. Moreover, the number of calls to the LMO routine will be similar to Theorem 1, with the LMO-approximation term now depending on the quality of the classifier learned at each step. We’ll include a discussion on this in the paper.

**Reviewer 4: Novelty.** While we agree that the paper combines ideas from prior works, our main contribution is the re-formulation of a constrained classification problem as an optimization problem over the intersection of two sets \(C \cap F\), and the novel application of results from Gidel et al. (2018) [13] to solve the resulting optimization. This allows us to provide a new learning algorithm which (i) has a simpler structure than the previous algorithm, (ii) enjoys better convergence rate, (iii) can better handle non-smooth constraints, and (iv) is more robust to choices of hyper-parameters.

Moreover, the proofs don’t directly follow from the previous papers for the following reasons: (i) Gidel et al. provide an optimization algorithm, which does not directly apply to a statistical ML setup. For example, their proofs assume an exact LMO, whereas we had to explicitly take into account the error due to finite sample, including in their so-called fundamental descent lemma. (ii) Gidel et al. only provide a bound on the duality gap for the constrained optimization problem; we convert this into a bound on the sub-optimality and infeasibility of the learned classifier.

Finally, we are able to handle a broader class of learning problems than Narasimhan (2018) [26], where the performance metrics can be defined by functions of more general “confusion vectors”, which can depend on the instance \(x\) in more intricate ways.

**Reviewer 6: Convexity.** We require the objective and constraints to be convex in the confusion matrix. We don’t see this as a strong requirement as it is satisfied by all the example metrics in Section 2, including common fairness metrics such as equal opportunity and equalized odds. Yes, \(B(u, r)\) is a ball of radius \(r\) centered at \(u\); \(\Delta_n\) is the \(n\)-dimensional simplex. We’ll make these notations clear.

**Reference**